# ***Java Script***

***Functions***

• (123.45).toFixed(5); //123.45000 (range of toFixed (0:100))

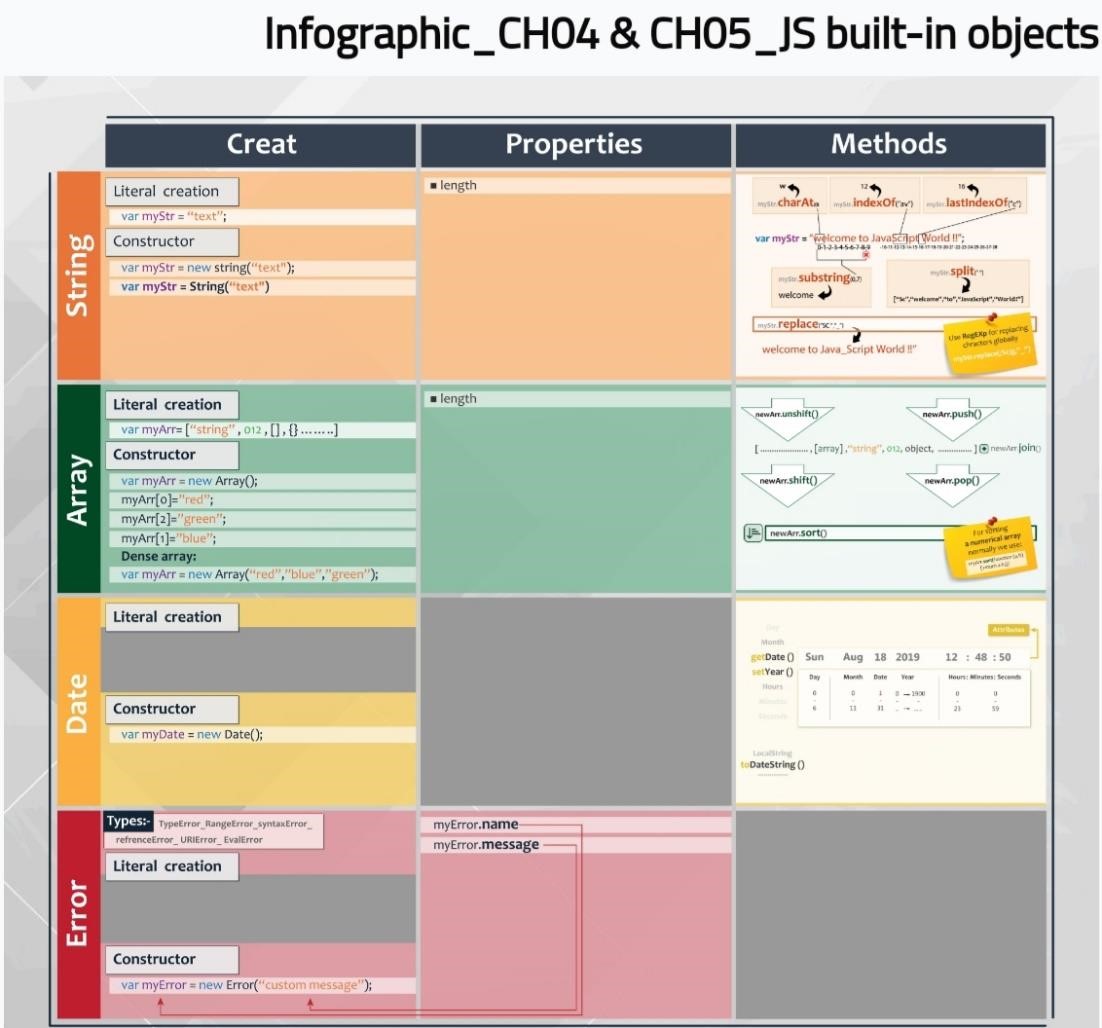
## Built-In Functions

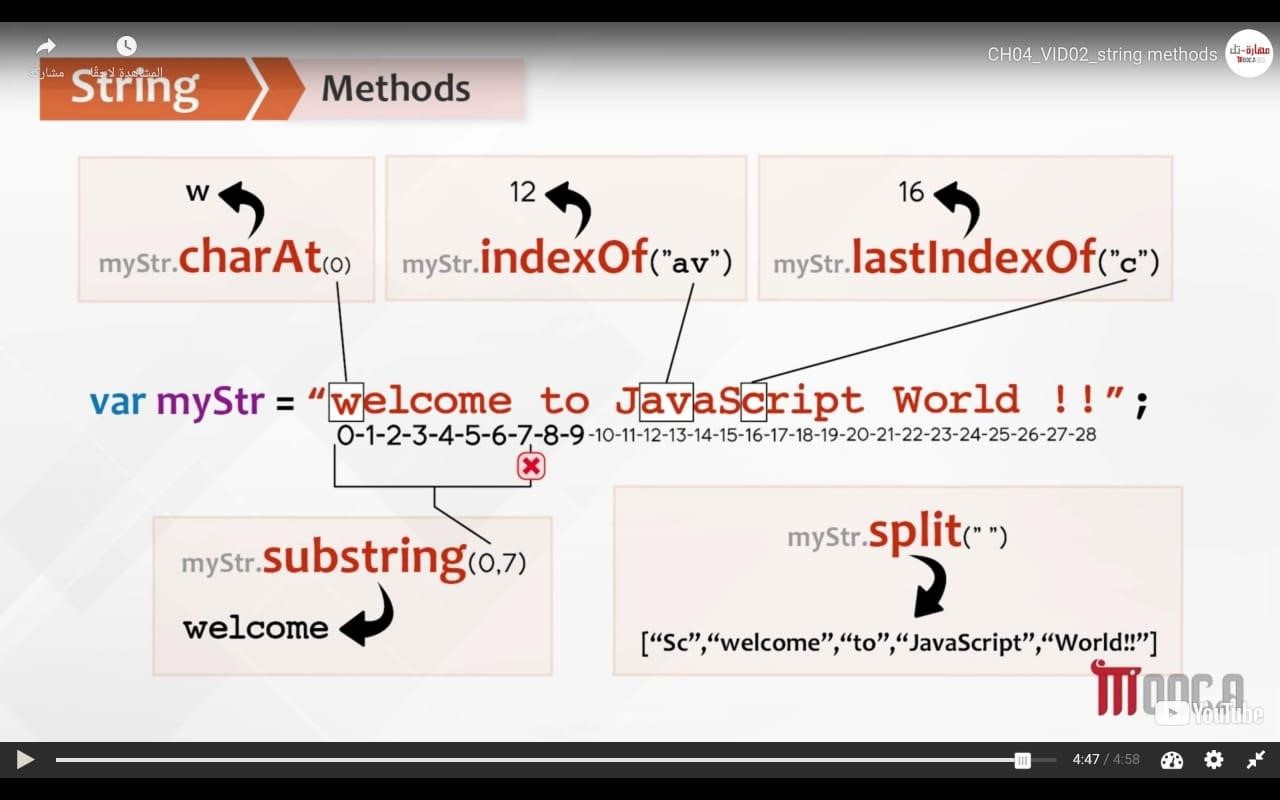
* ParseInt(x) -> return only number without (.) from string

(عمتا مش بتجيب الارقام اللى بعد الحرو ف) X = “ad1312” // NaN ▪

* + - X = “101abc” // 101
    - X = “101.01zcx” // 101
    - ParseInt(“0110” , “2(base)”) //6 -> convert from binary to number
* ParseFloat(x)
  + X = “101.01.23.ab” //101.01 -> to limit first (.) ▪ IsFinite(x)
  + x = "1152.3433.43"; // false
  + X = 123456 // true
  + X = “234567” //true
  + X = “1234fvb345” // false
  + IsNaN(x) -> inverse IsFinite(x)
  + X = 3456 //false
  + X = “3456dfg” // true
  + DecodeURL()
  + DecodeURLComponent()
  + EncodeURL()
  + encodeURLComponent()

## JS Built-In Objects



* String ex -> hello merOo
* Ways of writing string
  1. Str.repeat(number);
  2. Var str = “message”; //string
  3. Var str = new String(“message”); //object
  4. Var str = String(“message”); //string
* 
* Str.Length // 11
* Str.charAt(6) // m
* Str.indexOf(“e”) if not found will return –1 //1
* Str.lastIndexOf(“e”) //7
* Str.substring (1, 4) -> (اقص الاسترينج) // ell
  1. The end not print
* Split(“ ”) // [“hello” , “merOo”]
  1. Split(“”) // each character message.split(“s”); //[‘me’ , ‘ ’ , ‘age’]
* Replace ( /pattern/flag (i (ignore small or capital ), m (multiple lines) -> ( بدلى اول

, )حرف اللى كتباه بس g (global) ) , “”) o Replace(/e/ , “\_”) // h\_llo merOo o Replace(/e/ g , “\_”) // h\_llo m\_rOo o Replace(/o/ gi , “\_”) // hell\_ m\_r\_\_ o

String.fromCodePoint(Unicode , …);

* + String.fromCodePoint(65 , 66); // AB
* Array
* Length
* Dense Array -> var color = new Array (“red” , “blue” , “green”)
* Arr.join(“”); convert array to string

(اللى بيأثر فى الار ي) ▪

* 1. Arr.push(); addition in ending array arr.unshift(); addition in start array
  2. Arr.pop(); remove last element arr.shift(); remove first element
  3. Arr.reverse();
  4. Persons.splice(start, number remove); splice(2 , 1) -> meaning ((العنصر التانى فقط امسح
  5. Arr.some() , arr.every , arr.find()
* let arr = ['amira' , 'apple' , 'hassan' , 'yoka'];
* console.log(arr.every((val , idx , arr1)=>{
* return val[0] === 'a';
* })); // false
* let arr1 = ['amira' , 'apple' , 'hassan' , 'yoka'];
* console.log(arr1.some(( val , idx , arr1)=>{
* return val[0] === 'a';
* })); // true
* let arr2 = ['amira' , 'apple' , 'hassan' , 'yoka'];
* console.log(arr1.find((val)=>{
* return val === 'apple'; // apple -> if not found will print undefined
* }));
  1. MAB , FILTER , REDUCE
* arr = ['a' , 'd' , "5" , 1 , "6" , 2, 'c' , 'b'];
* arr1 = [];
* numbers = [11 , 5 , 9 , 3, 6];
* let fun = function(element){
* return element < 10;
* }
* console.log(numbers.map(fun)); //[false, true, true, true, true]
* console.log(numbers.filter(fun)); //[5, 9, 3, 6]
* console.log(numbers.reduce(fun , 10));   // true !??
* let fun1 = function(total , num){
* return total - num;
* }
* console.log(numbers.map(fun1));  // [11, 4, 7, 0, 2] // map(function(ele , index)) -> ele - index   NOTE -> (it is convert negative to positive)
* console.log(numbers.filter(fun1));  //  [11, 5, 9, 6]
* console.log(numbers.reduce(fun1 , 10)); //-24

( كل اللى موجود فى الاراى العاديه مش بيأثر عليها) >- Associative array ▪

لو كتب الانشيفت او البوش هيضاف فى الاول ولو حسبت اللينث هيحسب اللى ) >-NOTE 1.

( الكيي بتاعه ب صفر

* 1. Var ass = new Array ();
  2. Ass[“first”] = “mrmr”; first -> key , mrmr -> value
  3. For( i in ass){ console.log(ass[I]) }
* Date
* Var today = new Date(); //current date
  1. GET -> get information from date
     + Today.getDate(); //day (starting 1)
     + Today.getMonth(); //number of month (starting zero)
     + Today.getYear(); //year-1900 (ex-> if year = 2019 then output is

119) o Today.getFullYear(); //year

•

* 1. SET -> change date • Today.setDate(7); o Change day to 7
     + Today.setMonth(10); o Change month to November (11);
* because month starting with zero 3. Format

|  |  |  |
| --- | --- | --- |
|  | • | Today.toDateString(); //thu jun 15 2023 |
|  | • | Today.toLocaleString(); // 6/15/2023, 3:44:31 pm |
| ▪ | Error-Object 1.Types |  |
|  | • | Reference Error -> invalid reference is used ( بشاور على متغير مش  (موجود |
|  | • | Eval Error -> (بستخدم الfunctionبطريقه غلط) |
|  | • | URLError -> encodeURL() or decodeURL() are used incorrectly |
|  | • | Syntax Error -> ex (لما انسى احط اقوا س ) |
|  | • | Type Error -> ex (console.logg) error in word |
|  | • | RangeError -> (123.45).toFixed(101); because range of toFixed |

(0:100)

* Var myError = new Error(“Custom Error”); (كده مش هيظهر )

(كده هيظهر الكلمه بدل نوع الخطأThrow myError ( 1.

* 1. Throw “message”; //uncaught message
  2. Throw new RangeError(“message”); (call RangeError ) //uncaught RangeError: message
  3. Console.log(myError.name); //Error
  4. Console.log(myError.message); // Custom Error
  5. Statement after error doesn`t execute
* JS Error Handling
  1. Try{} … catch(e){} … finally{}
     + - [try ... catch](https://fcihelwanedu-my.sharepoint.com/personal/amira_20210185_fci_helwan_edu_eg/Documents/Transcribed%20Files/Your%20Recording%2017.wav)

اول ما يلاقى خطأ فى تراى هيعمل اسكيب لباقى التراى ويروح على الكاتش ولو ) •

( فى حاجه تتنفذ بعد الكاتش هيروحلها

* + - * Statement of catch only execute if exit error

(لو فى خطأ فى الكاتش هيظهر الخطأ ومش هيظهر اخر جمل ه) •

بيتكتب بعد الكاتش علشان لو فيها خطأ هيظهر الخطأ عادى وهيطبع ) >-Finally •

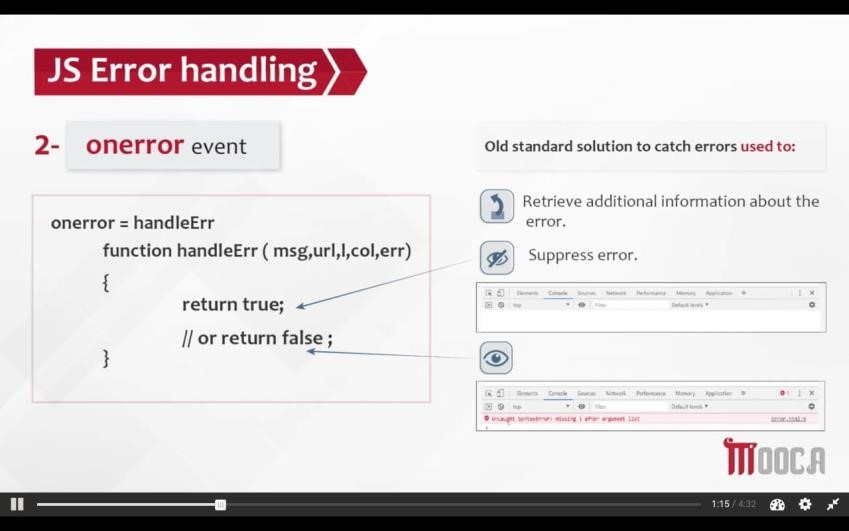
(اللى جوه الفاينلى

* + - * [catch(e)](https://fcihelwanedu-my.sharepoint.com/personal/amira_20210185_fci_helwan_edu_eg/Documents/Transcribed%20Files/Your%20Recording%2018.wav)
      * Catch(e) { if (e instanceof RangeError) {} else if (e == “message”)

{} }

•

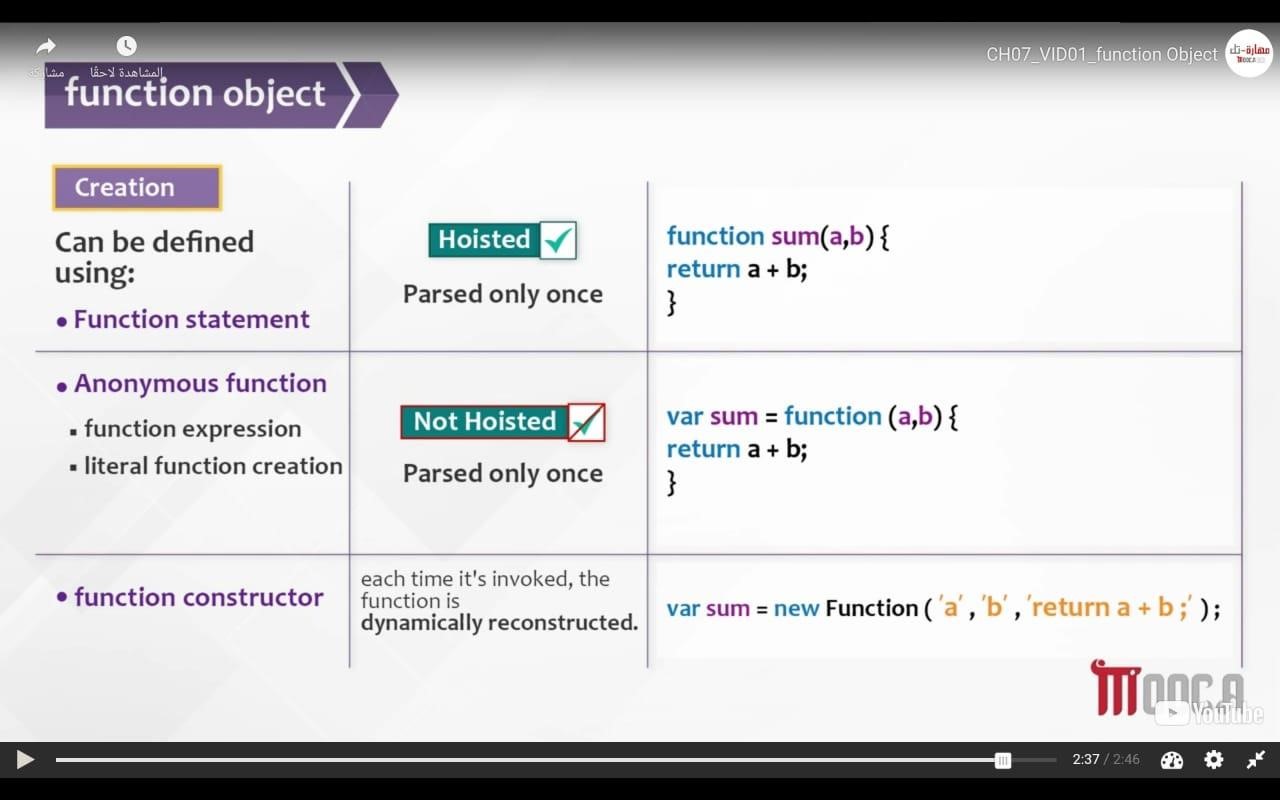
2. Onerror event

* + - * [OnError event](https://fcihelwanedu-my.sharepoint.com/personal/amira_20210185_fci_helwan_edu_eg/Documents/Transcribed%20Files/Your%20Recording%2019.wav)
      * 
* Function Object
* IIFE -> Immediately Invoked Function Expression (بنادى عليها مره واحده بس وبتبقى ملهاش اسم)
* (function(){
* console.log("hii");
* })()
* //problem
* function outer(){
* var arr = [];
* for(var i=0; i<3; i++){
* arr.push(function(){
* console.log(i);
* })
* }
* return arr;
* }
* var result = outer();
* result[0](); //3
* result[1](); //3
* result[2](); //3

▪

* //solution using IIFE Pattern
* function outer(){
* var arr = [];
* for(var i=0; i<3; i++){
* arr.push((function(){
* console.log(i);
* })());
* }
* return arr;
* }
* var result = outer();
* result[0]; //0
* result[1]; //1
* result[2]; //2

▪

* 
* Arguments

(علشان اعرف الفانكشن هيجيلها كام باراميترArguments.length ( 1.

* + - * + function sum(){
        + var result = 0;
        + for(var i=0; i<arguments.length; i++){
        + result += arguments[i];
        + }
        + return result;
        + } •

* + - * + console.log(sum(1 , 4 , 5));

•

2.

* Trying use join with string ex var str = "this ia JavaScript"; //function borrowing using apply

console.log([].join.apply(str , ["\*\*"]));

//t\*\*h\*\*i\*\*s\*\* \*\*i\*\*a\*\* \*\*J\*\*a\*\*v\*\*a\*\*S\*\*c\*\*r\*\*i\*\*p\*\*t

* + - 1. //function borrowing using call method console.log([].join.call(str , "\*\*"));

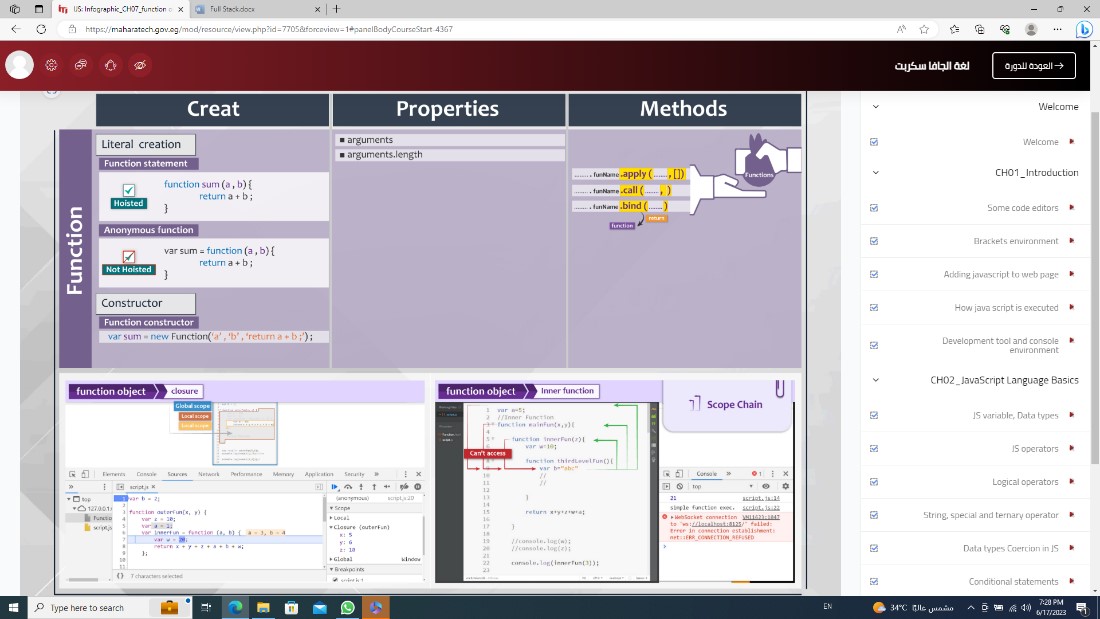
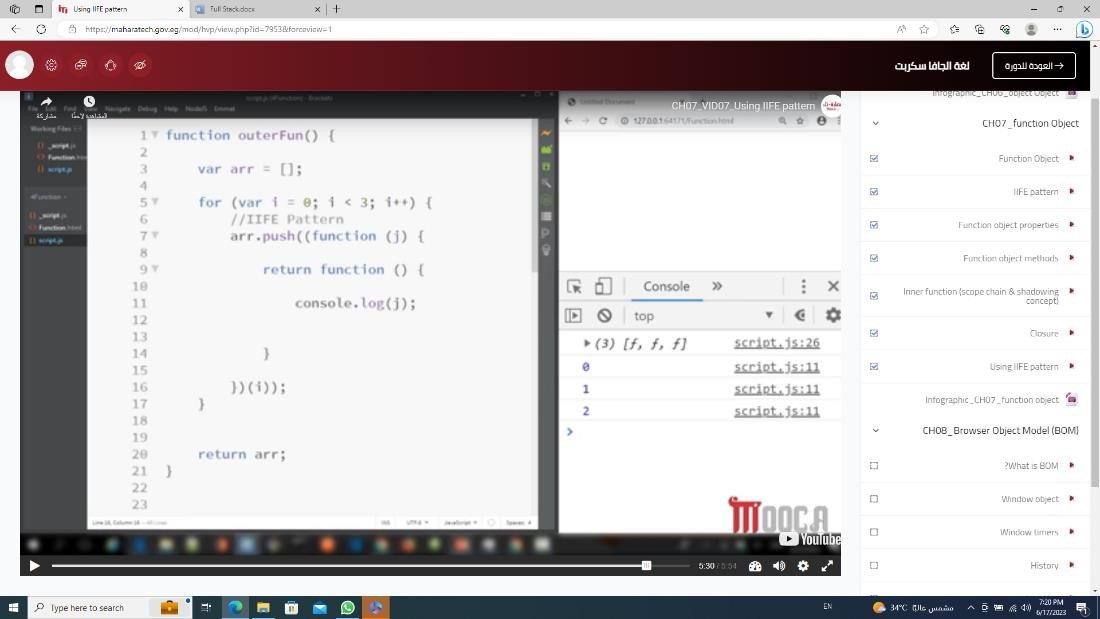
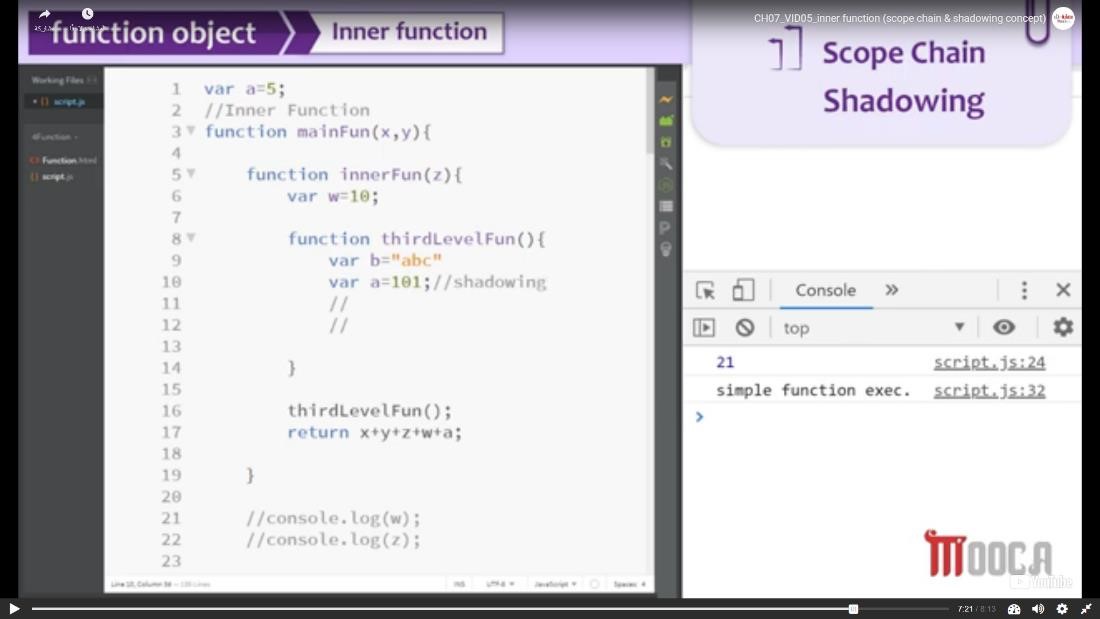
//t\*\*h\*\*i\*\*s\*\* \*\*i\*\*a\*\* \*\*J\*\*a\*\*v\*\*a\*\*S\*\*c\*\*r\*\*i\*\*p\*\*t

* + - 1. //binding

var result = [].join.bind(str , "\_\_"); console.log(result("\*\*"));

//t\_\_h\_\_i\_\_s\_\_ \_\_i\_\_a\_\_ \_\_J\_\_a\_\_v\_\_a\_\_S\_\_c\_\_r\_\_i\_\_p\_\_t

▪



* Object
* Instant method o Obj.hasOwnProperty(“name of key”); //true or false o Obj.toString(); //[object Object]
* Static method (Class method)
* o Object.steal(obj);
* o Object.freaze(obj);
* o Object.keys(obj);
* o Object.values(obj);
* o delete obj.key of obj; o Use of For (in)

Object.defineProperty(obj , “name of property ”, {})

// To each property

* Value -> default(undefined) -> value of property
* Writable -> default (false) -> Don`t allow to set a new value
* Configurable -> default (false) -> Don`t allow to delete property
* Enumerable -> default (false) -> property will not appear in for(in)
* Set : function(val){ variableName = val} to change value
* Get : function(){return variableName} to see the value
* var obj = {"myName" : "amira" , "age" : "19"};
* Object.defineProperty(obj , "myName" , {
* value : "amira",
* writable : false,
* configurable : false,
* enumerable : false
* })

▪

* for(i in obj){
* console.log(obj[i]); // 19
* }
* obj.myName = "meroo";
* console.log(obj.myName); // amira

▪

* delete obj.myName;
* console.log(obj.myName); // amira
* Object.defineProperties(obj , { property:{}, property:{} })
* var obj = {"myName" : "amira" , "age" : "19"};
* Object.defineProperties(obj , {
* myName:{
* value : "amira",
* configurable : false
* },
* age : {
* value : "19",
* writable : true
* }
* })
* delete obj.myName;
* console.log(obj.myName); //amira

▪

* obj.age = "20";
* console.log(obj.age); //20
* NOTE ->

• Data Descriptor contains (value, writable) • Accessor Descriptor contains (set. get)

var myName = "amira"; var age = "19"; var obj = {};

Object.defineProperties(obj , {

myName:{

get : function(){return myName;}, set : function(val){myName = val;}, configurable : false

},

age : {

value : "19", writable : true

} })

obj.myName = "meroo";

console.log(obj.myName); //amira

// Factory Method

* function employee(nm , dep , sal){
* return {
* empNumber : nm,
* empSalary : sal,
* empDep : dep,
* info : function(){
* return "employee works in dept " + this.empDep +

"and empNumber is " + this.empNumber;

* }
* }
* }

▪

* console.log(employee("amira" , "it" , "100000")['info']());

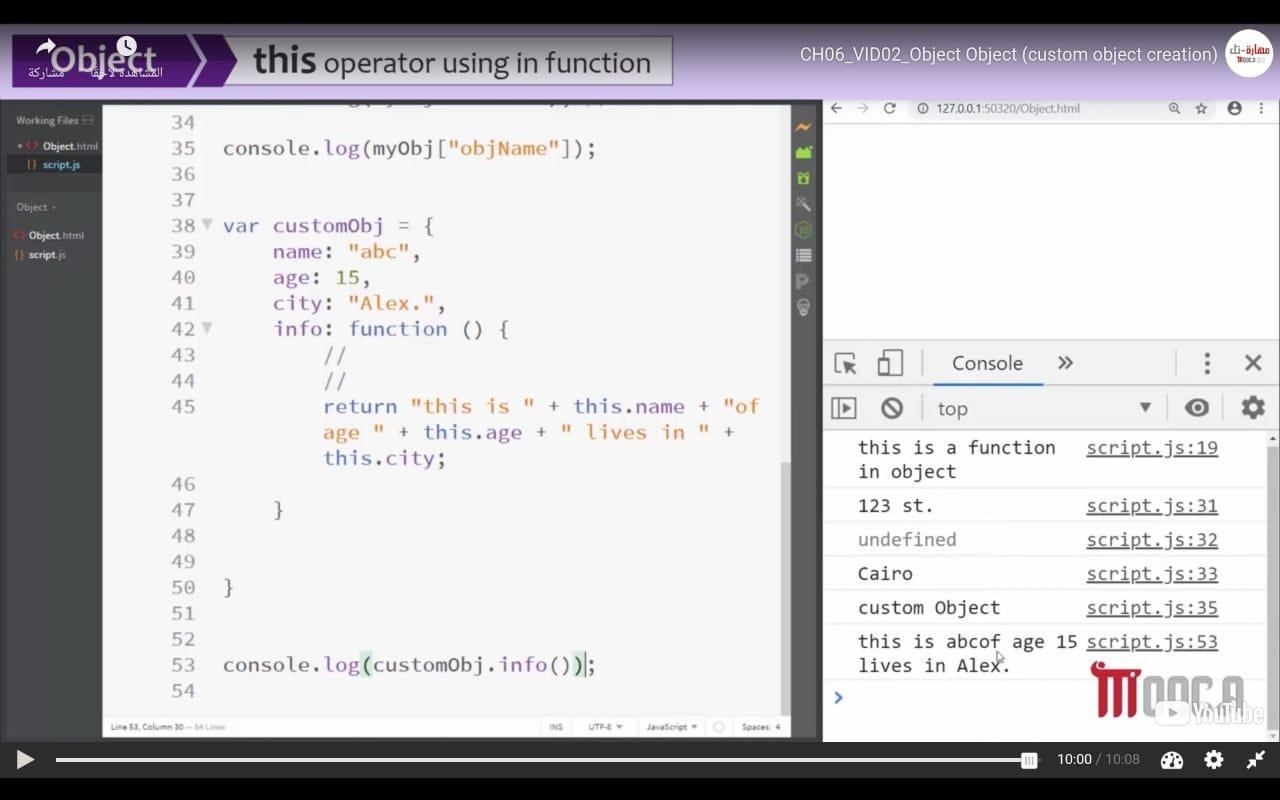
▪

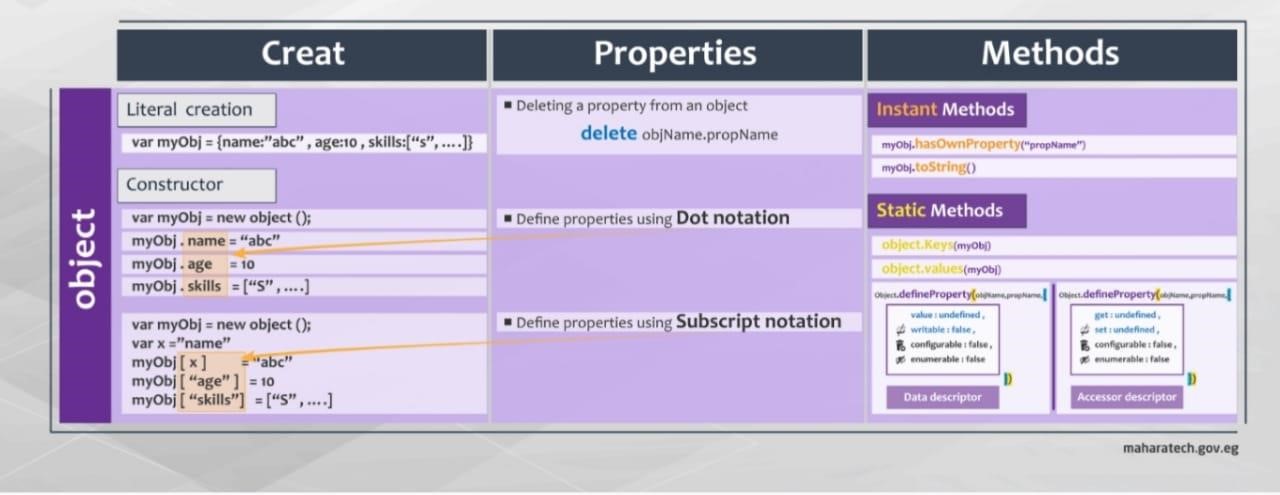
* // Constructor Method
* function Employee(nm , dep , sal){
* this.empNumber = nm;
* this.empSalary = sal;
* this.empDep = dep;
* }

▪

* var emp = new Employee("amira" , "it" , "10000");
* console.log(emp.empNumber);

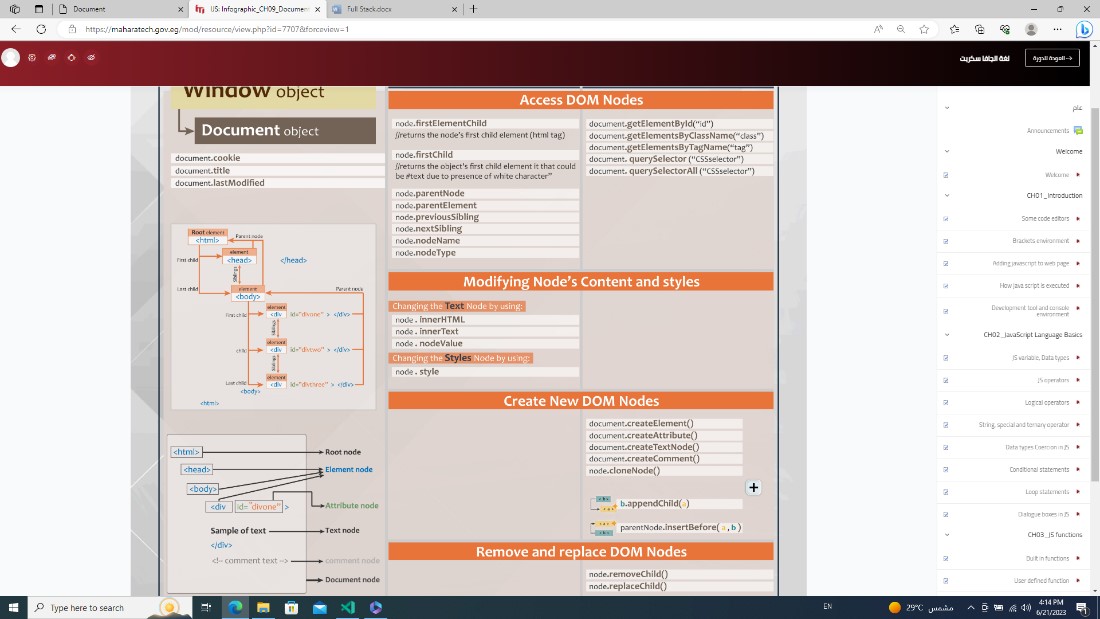
o





NOTE -> Dot notation objName.property = value Subscript Notation objName[“property”] = value

## BOM -> (Browser object model)

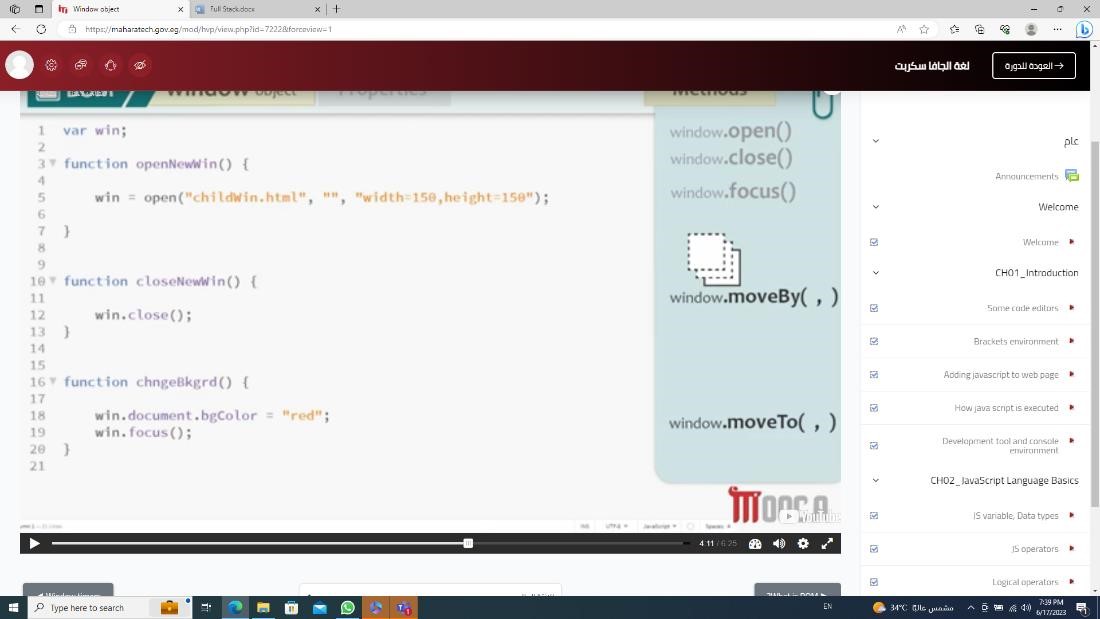


* Window
  + - Alert()
    - Confirm() return true or false o Prompt(“” , default) o setTimeout(fun , duration) -> (appear only once )& clearTimeout()

var stop; function start(){ console.log("hiii merOoo"); stop = setTimeout(start , 2000); }

function end(){ clearTimeout(stop);

}

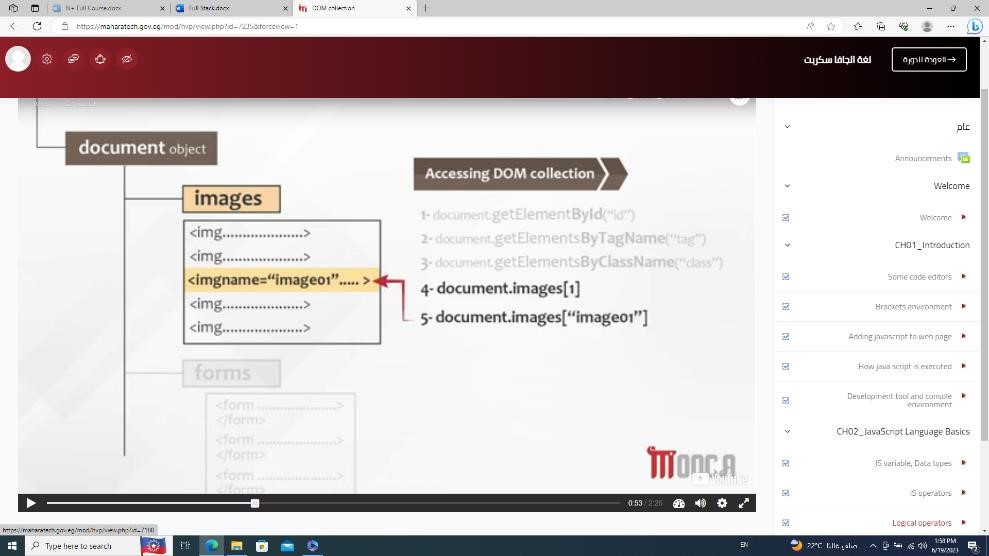
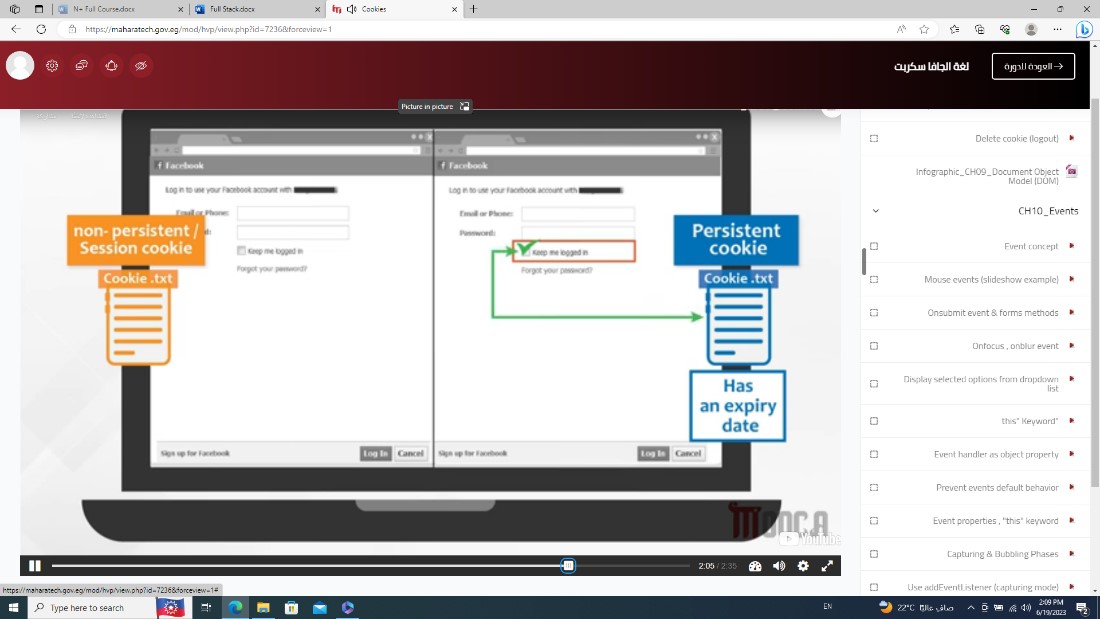
* + - SetInterval(fun , duration) & clearInterval(name variable of setInterval)
      * + var stop;
        + function start(){
        + stop = setInterval(function(){
        + console.log("hello merooo");
        + } , 2000)
        + }
        + function end(){
        + clearInterval(stop);
        + }
      * Open(“url” , “name or target(blank or self)” , “place with css (width:100 , height:200)”)
      * Close() o Stop() o Print()
      * Focus()
      * scrollTo(x , y , {left:10 , top:30 , behavior:”smooth”}) o
      * scrollBy() o onscroll o .resizeBy() o .resizeTo()
      * 

* History (count tab)
  + .length
  + .go(negative to back or positive to forward)
  + .forward()
  + .back()
* Location
  + .href

o .protocol

* + .host or .hostname
  + hash
  + .replace(“url”) -> replace in history
  + .assign(“url”) -> (remove all forward and reach to page of url) o .search -> (to get query from form) o Ex -> (?username=Ali found in url)
  + .reload()
* Navigator (language related to browser) o .appCodeName -> (to know name of browser but it isn`t clear) o UserAgent -> (to know name of browser as string and clear )
  + .cookieEnabled -> (to know, if user allow put cookie) o .language
  + .platform -> (to know operating system that client use it)
* localStorage & sessionStorage (it is exit if you duplicated the tap) o key(index) o // SET
  + .setItem(“name” , “value”) o .”name” = “value” o [“name”] = “value” o // GET
  + .getItem(“name”) o .name o [“name”] o REMOVE
  + removeItem(“name”) o clear();

# // DOM()

* Document (contain all HTML) o Console.dir(document) -> to view DOM prosperities o .body.children[0].nextElementSibling o // Ways to select element
  + -  o .getElementById(“”) o .getElementsByClassName(“”)
    - .getElementsByTagName(“”)
    - .querySelector(‘#id or .class’); o .querySelectorAll(‘’);
    - .innerHTML = “”; o .innerText = “”;
    - (call div, p or ...).className -> return name of class o (call div, p or ...).classList.add(“name of new class”) o (call div, p or ...).classList.remove(“name of new class”) o GetComputedStyle(call div, p or ...).color // (بسأل لونه ايه وهكذا مع باقى ) o Body.insertBefore( ,اللى عايزه احطه b )
    - .appendChild(); o CreateElement(); o CreateAttribute(); o CreateTextNode(); o CreateComment();
* Cookie
  + Persistent -> has an expire date o None-persistent(session)
  +  o

## • Events

*  o Onmouseover(); o Onmouseout(); o .onmousemove
* document.getElementById('move').onmousemove = function move (e) {
* document.getElementById('move').innerHTML =

e.clientX;

* }
* e.clientX (from starting page) o e.offsetX (start from element ) o e.target (info about event)
* e.type (type of event)
* FORM o Onsubmit

### ▪ JS

* document.forms[0].onsubmit = function(){
* alert(document.getElementById('inp').value);
* }

▪ HTML

<form action="">

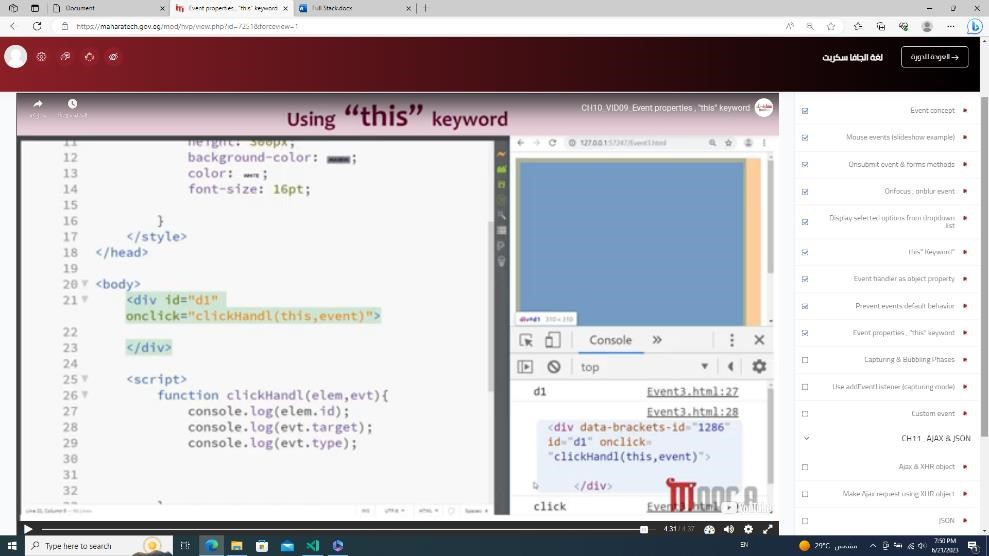
<input type="text" value = "hii" id="inp" placeholder="name">

<button type="submit">submit</button> </form>

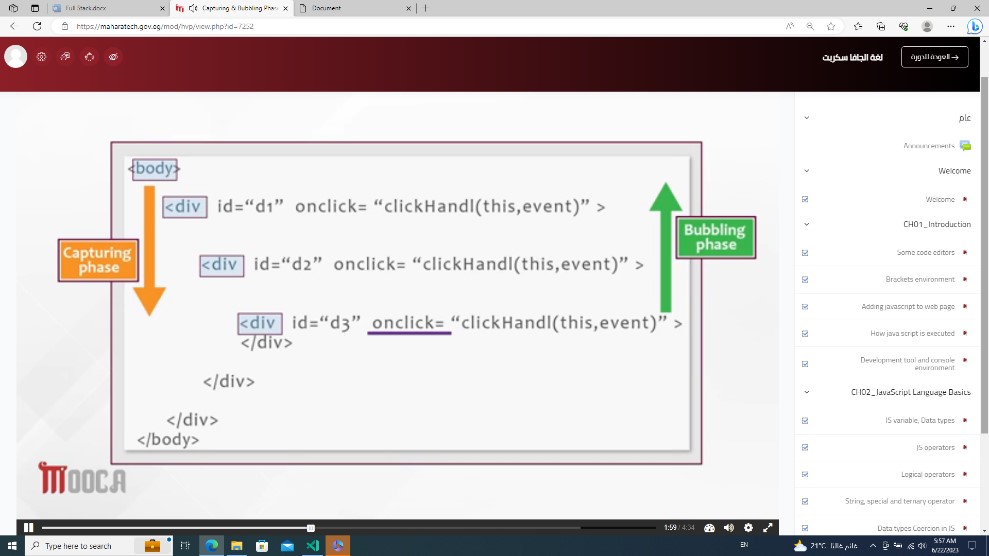
o Onreset o Onfocus o Onblure

• <input type="text" value = "hii" id="inp" placeholder="name" onblur="this.value='nothing'">

### • o

* Dropdown list o .length
* .selectedIndex (return a number of option who user choose it)
* .options[] (return all options who user choose it)
  + .options[].selected
  + .options[].text
  + .options[].value o
* THIS
* 

o

* 
* Event.stopPropagation()
* AddEventListener(“click” , function(){} , true or false)
* True -> capturing mode (لو استخدمتها مع stopPropagation هيظهر اول div ب س)
* False -> bubbling mode

نءئىؤ ن o

To Create Event

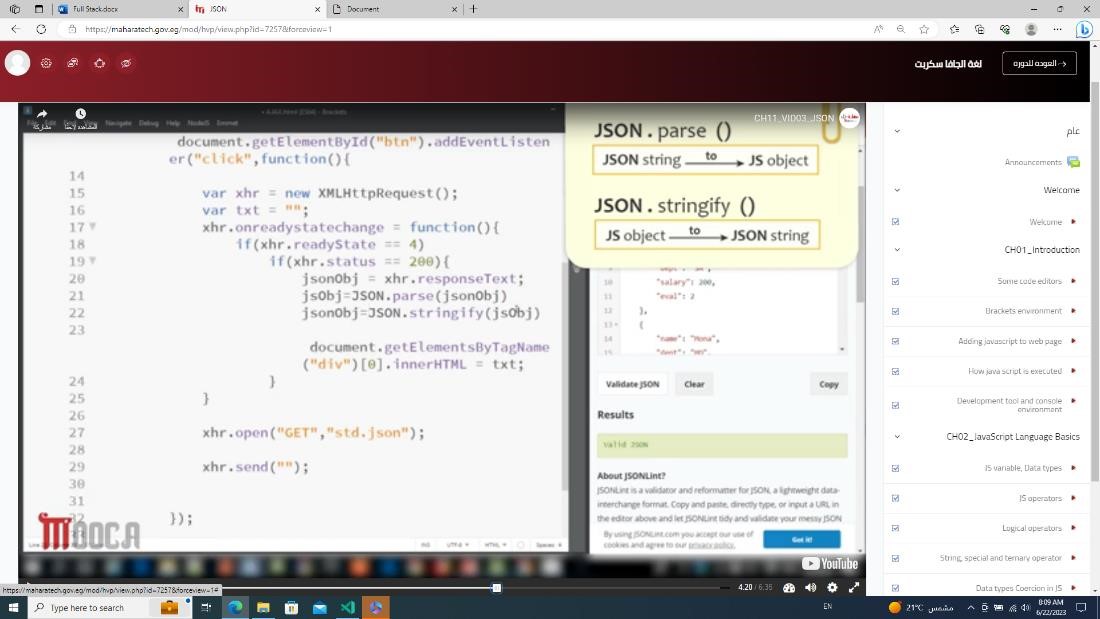
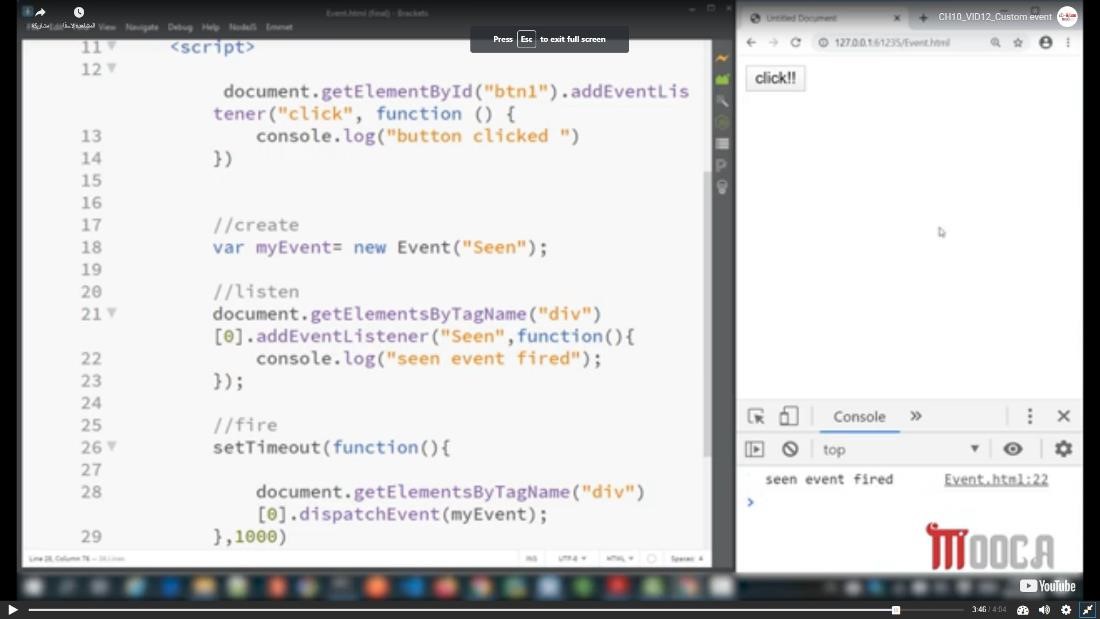
* //create
* var newEvent = new Event("seen");

•

* // listen
* document.getElementsByTagName("div")[0].addEventListener("s een" , function(){
* console.log("seen Event fired");
* }); •

* // fire
* setTimeout(function(){
* document.getElementsByTagName("div")[0].dispatchEvent(newEv ent);
* } , 1000); •

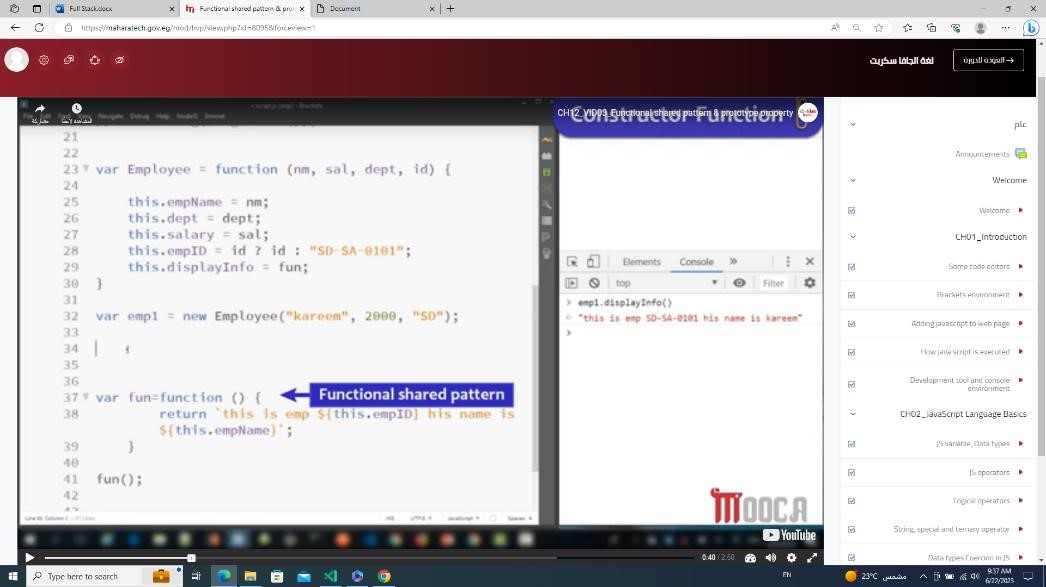
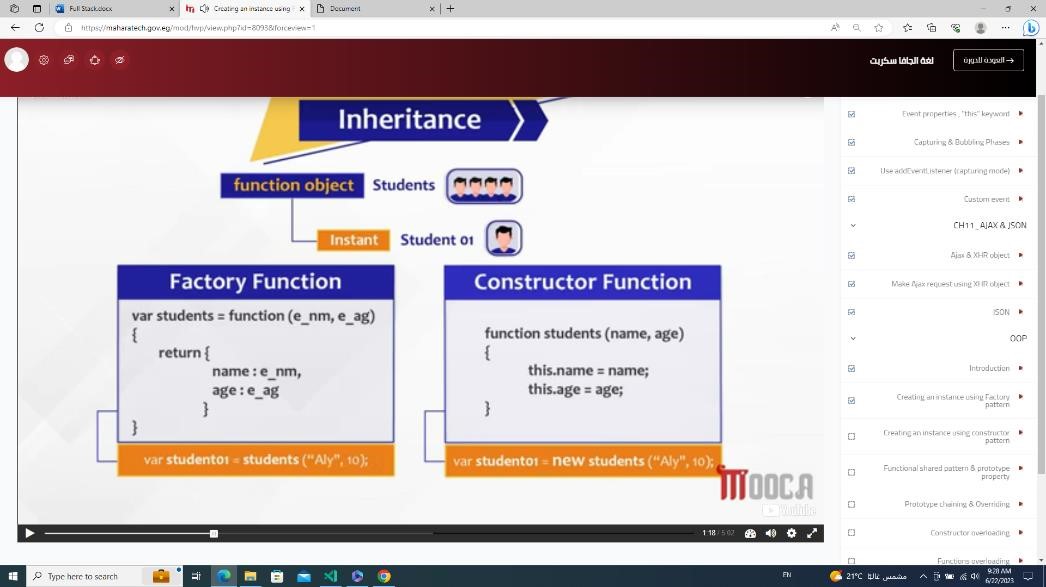
•

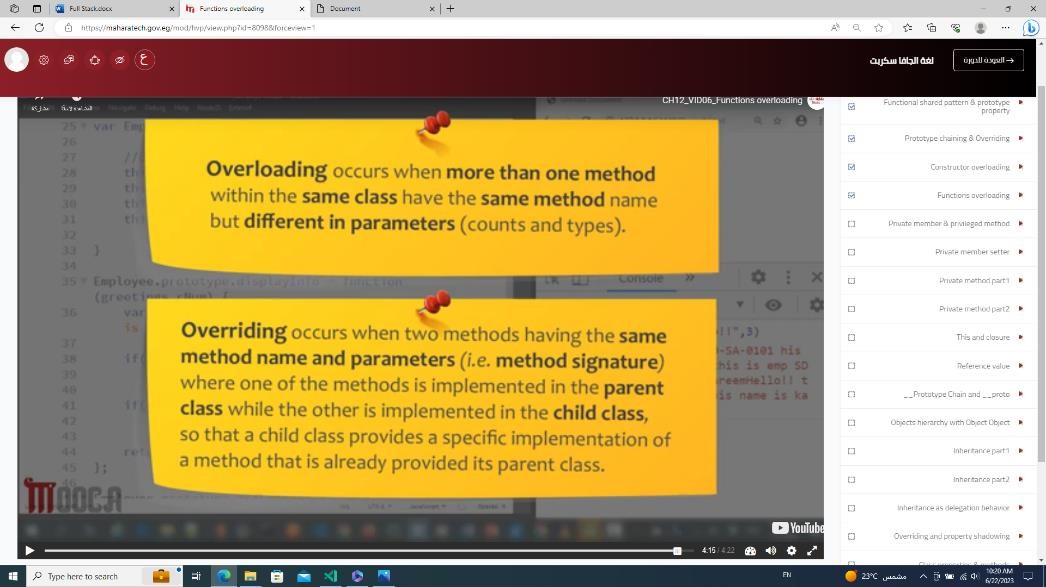


•

## AJAX ->

o



o o 

***JSON***

* JSON.parse(json name); //convert JSON into Object
* JSON.stringify(myObject); // convert Object into JSON

***AJAX -> (Asynchronous JavaScript and Xml)***

* XHR object -> new XMLHttpRequest
  + Properties
    - .onreadystatechange
    - .readyState
      * (0 -> uninitialized, 1-> loading, 2-> loaded, 3-> interactive, 4-> complete)
    - .status (return statue as a number)
      * If start from (1 : Information , 2:Success , 3:Redirection , 4: Client Error , 5: Server Error)
    - .statusText (return status as a string)
    - .responseText (return the response as a String)
  + Methods
    - Open(“method(get or post)” , “URL” , “userName” , “password”)
    - Send(“content (if POST) ”)

***########## ES6 (Native JS)#################***

***A screenshot of a computer

Description automatically generated***

***Let & var & const***

* Let -> block scope & not hoist
* Var -> function scope & hoist
* Const -> it must initialized & block scope & not hoist

***Assign object to another object without change the value***

Obj2 = {…obj} // obj2.name = “MerOoo”; obj.name = “not change”

***Set([])***

* ***Property***
  1. ***size***
* ***Methods***
  1. ***add()***
  2. ***delete()***
  3. ***Has()***
  4. ***Clear()***
* ***Iterator Methods***
  1. ***keys()***
  2. ***entries()***
  3. ***next()***

let s = new Set([1 , 2, 1,4, 5, 1]);

console.log(s.entries()); // {1 => 1, 2 => 2, 4 => 4, 5 => 5}

console.log(s.size); // 4

console.log(s.add(88)); // Set(5) {1, 2, 4, 5, 88}

console.log(s.delete(1));  // true

console.log(s); // Set(4) { 2, 4, 5, 88}

console.log(s.keys().next()); // {value: 2, done: false}

console.log(s.keys().next().value); // 2

console.log(s.keys().next().done); // false -> means set doesn`t finish

s.forEach(element => {

        console.log(element); // 2 4 5 88

});

***WeakSet***

// let ws = new WeakSet([{"1":2 , "4": 55 , "num" : 88}]);

// console.log(ws);

***Map([[] , [] , [] , …])***

* ***Property***
  1. ***size***
* ***Methods***
  1. ***set()***
  2. ***Get()***
  3. ***delete()***
  4. ***Has()***
  5. ***Clear()***
* ***Iterator Methods***
  1. ***keys()***
  2. ***entries()***
  3. ***values()***

let mm = new Map();

mm.set("10" , "string");

mm.set(10 , "number");

console.log(mm);   //Map(2) {'10' => 'string', 10 => 'number'}

console.log(mm.get(10)); //number

console.log(mm.get("10")); //string

console.log(mm.size); // 2

console.log(mm.delete(10));  // true

console.log(mm); // Map(1) {'10' => 'string'}

console.log(mm.clear());  // undefined

console.log(mm); // Map(0) {size: 0}

***WeakMap***

let wm = new WeakMap();

wm.set({name : "amira"} , "sscsc");

console.log(wm)

***for(I of arr)***

let arr = ['q' , 5 , 'd' , '7'];

for(i of arr){

        console.log(i); // q 5 d 7

}

let obj = {

        's' : 5,

        3 : 11,

        9 : 'uu'

};

for(i of obj){

        console.log(i); // Uncaught TypeError: obj is not iterable

}

***With object like map***

let mm = new Map([[2 , 'a'] , [5 , 'd']]);

console.log(mm);

for(i of mm){

        console.log(i); // [2 , 'a']  [5 , 'd']

}

for(i of mm.keys()){

        console.log(i); // 2 5

}

for(i of mm.values()){

        console.log(i); // a d

}

for([k , v] of mm){

        console.log(`${k} : ${v}`); // 2 : a  5 : d

}

***Destructor***

let arr = [2,3, , 'a' , 'b','g' , 9, 3];

let [x , ,y = 10 , , z = 'n' ,  t , r] = arr;

console.log(t); // g

console.log(r); // 9

console.log(y); // 10

console.log(z); // b

…arr1

let arr = [2,3, , 'a' , 'b','g' , 9, 3];

let [x , ,y = 10 , ...arr1] = arr;

console.log(y); // 10

console.log(arr1); // ['a' , 'b','g' , 9, 3]

Object

let obj ={

        name : "ES6",

        main : "js",

        born : 2015,

        newObj : {

                country : "cairo",

                nat : "Egypt"

        }

}

//  let {name:x , main:y , born, zz = "default" , newObj:{

//         country:coo , nat : naa}} = obj;

let x , y;

({name:x , main:y , born , zz = "default" , newObj:{

        country:coo , nat : naa

}} = obj);

console.log(x);  // ES6

console.log(born); // 2015

console.log(zz); // default

console.log(coo); // cairo

console.log(naa); // Egypt

***Fetch API***

fetch("test.json")

.then((result)=>{

        console.log(result);  // Response {}

        let jsObject = result.json();

        console.log(jsObject); // promise{}

        return jsObject;

}).then((jsObject) => {

        jsObject.length = 2;

        return jsObject;

}).then((jsObject)=>{

        console.log(jsObject['name']); // dfsghdj

});

***Promise***

* Pending : initial State
* FulFilled : Complete Successfully
* Rejected : Faild
* const myPromise = new Promise((resolveFunction , rejectedFunction) => {
* let condition = true;
* if(condition){
* resolveFunction(`You Are Good ? `);
* } else{
* rejectedFunction(`You Are Bad ? `);
* }
* })
* // .then(
* //         (resolveValue) => console.log(`${resolveValue} Good`),
* //         (rejectedValue) => console.log(`${rejectedValue} bad`)
* // )       // You Are Good ?  Good
* console.log(myPromise);
* myPromise.then(
* (resolveValue) => console.log(`${resolveValue} Good`),
* (rejectedValue) => console.log(`${rejectedValue} bad`)
* )
* myPromise.then(
* (resolveValue) => console.log(`${resolveValue} Good`),
* (rejectedValue) => console.log(`${rejectedValue} bad`)
* )
* ***Catch(appear in case error) & finally (always run first)***
* let myPromise = new Promise((resolveFunction , rejectFunction)=>{
* let names = ["amira" , "aya" , "rokaa" , "rana"];
* if(names.length == 4){
* resolveFunction(names);
* } else{
* rejectFunction(Error("Number Of Employes Is Not 4"));
* }
* });
* myPromise
* .then(
* (resolve) => {
* resolve.length = 2;
* console.log(resolve);
* }
* )
* .then(
* (resolve) => {
* resolve.length = 2;
* console.log(resolve);
* })
* .catch(
* (reject) => console.log(`${reject}`)
* )
* .finally(
* console.log("The process is done ")
* )
* // The process is done (2) ['amira', 'aya'] TypeError: Cannot set properties of undefined (setting 'length')
* Promise.all([])
  + If All res will print in console -> all values in resolve
  + If one of them rej will print it
* const firstPromise = new Promise((res , rej) => {
* setTimeout(()=>{
* res("Iam the first Promise");
* } , 1000);
* });
* const secondPromise = new Promise((res , rej)=>{
* setTimeout(()=>{
* rej(("Iam the second Promise"));
* } , 1000);
* });
* const thirdPromise = new Promise((res , rej)=>{
* setTimeout(()=>{
* res(("Iam the third Promise"));
* } , 1000);
* });
* Promise.all([firstPromise , secondPromise , thirdPromise]).then(
* (resolve)=>{
* console.log(resolve);
* },
* (reject)=>{
* console.log(reject);
* }
* ) // Iam the second Promise
* Promise.allSettled([]) return all promise with status if res or rej
* Promise.allSettled([firstPromise , secondPromise , thirdPromise]).then(
* (resolve)=>{
* console.log(resolve);
* },
* (reject)=>{
* console.log(reject);
* }
* )/\* (3) [{…}, {…}, {…}]
* 0: {status: 'fulfilled', value: 'Iam the first Promise'}
* 1: {status: 'rejected', reason: 'Iam the second Promise'}
* 2: {status: 'fulfilled', value: 'Iam the third Promise'}
* length: 3
* [[Prototype]]: Array(0)
* \*/
* Promise.race([]) return first promise will run
* Promise.race([firstPromise , secondPromise , thirdPromise]).then(
* (resolve)=>{
* console.log(resolve);
* },
* (reject)=>{
* console.log(reject);
* }
* ) // Iam the first Promise
* Async can be function to return promise
* async function getData(){
* let arr = ['MorAa'];
* if(arr.length > 0){
* return  "Users Found";
* } else{
* return Error("Users Not Found");
* }
* }
* console.log(getData());  // Promise {<fulfilled>: 'Users Found'}
* getData().then(
* (res)=>{
* console.log(res);
* },
* (rej) => {
* console.log(rej);
* }
* ) // Users Found
* Await -> stop program until print promise
* const myPromise = new Promise((res , rej)=>{
* setTimeout(()=>{
* res("Iam Promise");
* } , 2000);
* });
* async function getData(){
* console.log("before promise");
* console.log(await myPromise);
* console.log("after promise");
* }
* getData();
* /\*
* before promise
* Iam Promise
* after promise
* \*/

Reject

* const myPromise = new Promise((res , rej)=>{
* setTimeout(()=>{
* rej(Error("Iam Promise"));
* } , 2000);
* });
* async function getData(){
* console.log("before promise");
* await myPromise.catch((reject)=>{
* console.log(reject);
* });
* console.log("after promise");
* }
* getData();
* /\*
* before promise
* Error: Iam Promise
* at learn.js:426:21
* after promise
* \*/

***Generator***

* Function \*generatorName(){yield num;}
  + (لما ادخل على اليلد اللى بعدها هينفذ امر بتاعها)
* function \*generateNumber(){
* yield 1;
* console.log("Yield 1");
* yield 2;
* console.log("Yield 2");
* yield 3;
* console.log("Yield 3");
* yield 4;
* console.log("Yield 4");
* }
* function \*generateLetters(){
* yield "a";
* console.log("Yield a");
* yield "b";
* console.log("Yield b");
* yield "c";
* console.log("Yield c");
* }
* function \*generatorAll(){
* yield \*generateNumber();
* yield \*generateLetters();
* yield\* [10 , 20 , 'D' , 30];
* }
* let generate = generateNumber();
* console.log(generate.next());    // {value: 1, done: false}
* console.log(generate.next());  //Yield 1 // {value: 2, done: false}
* for(let value of generate){
* console.log(value); //Yield 2 3  Yield 3 4  Yield 4
* }
* console.log("###############");
* let generateAll = generatorAll();
* for(let value of generateAll){
* console.log(value); //1 Yield 1 2 Yield 2 3 Yield 3  4 Yield 4 a Yield a b Yield c Yield c 10 20 D 30
* }

***Modules***

* Index.html
  + <script src="learn.js" type="module"></script>
  + <script src="imp.js" type="module"></script>
* Learn.js
  + let x = 10;
  + let arr = [1 , 2, 3, 4];
  + function sayHello(){
  + return `Hello MoRaaa`;
  + }
  + export { x , arr , sayHello};
* Imp.js
  + import {x , arr , sayHello} from "./learn.js";
  + console.log(x);
  + console.log(arr);
  + console.log(sayHello());

***Symbol -> dataType***

let obj = {

        [Symbol.for(10)] : "ab",

        [Symbol(10)] : "sss",

        nm : "Mora"

}

console.log(obj[Symbol(10)]); // undefined

console.log(obj[Symbol.for(10)]); // ab   NOTE -> should use for with Symbol to get value

for(i in obj){

        console.log(`${i} : ${obj[i]}`); // nm : Mora

}

console.log(Object.getOwnPropertySymbols(obj));  // (2) [Symbol(10), Symbol(10)]   NOTE -> in for loop the Symbol is hidden so use of getOwnPropertySymbols

let y = Symbol.for(88);

console.log(y); // Symbol(88)

console.log(Symbol.keyFor(y)); // 88

console.log(y.description);  // 88

***BigInt***

let x = BigInt(10);

console.log(`${x} -> dataType -> ${typeof x}`); // 10 -> dataType -> bigint

let y = 20n;

console.log(`${y} -> dataType -> ${typeof y}`); // 20 -> dataType -> bigint

let num = 5;

// console.log(x + num);   // Cannot mix BigInt and other types, use explicit conversions

console.log(num + parseInt(x)); // 15  NOTE BigInt can convert into Number with using parseInt alone

console.log(BigInt(num) + x);  // 15n

// console.log(num + (+x)); // Cannot convert a BigInt value to a number

***Nullish Operator***

let x = 10;

console.log(x || 20); // 10

console.log(x && 20); // 20

console.log(x ?? 20); // 10

console.log("#######");

x = 0;

console.log(x || 20); // 20

console.log(x && 20); // 0

console.log(x ?? 20); // 0

console.log("#######");

x = undefined; // OR NULL

console.log(x || 20); // 20

console.log(x && 20); // undefined

console.log(x ?? 20); // 20

let obj = {

        num : 10,

        age : 20,

        name : "MoRaa",

};

console.log(obj.num.x); // undefined

// console.log(obj.num.x.v); // Cannot read properties of undefined (reading 'v')

console.log(obj.num.x?.v); // undefined

// console.log(obj.num.abc()); // obj.num.abc is not a function

console.log(obj.num.abc?.()); // undefined

***OOP***

* User.prototype
* User.prototype.sayHello = function(){}
* Object.prototype.sayHello = “hello MoRaa”
  + Add sayHello to all objects
* class User{
* static count = 0;
* constructor(name , age , salary){
* this.n = name;
* this.a = age;
* this.s = salary;
* this.msg = function(){
* return `hello ${this.n} `;
* }
* User.count++;
* }
* add(amount){
* this.s += amount;
* }
* }
* class Admin extends User{
* constructor(name , age , permission){
* super(name , age);
* this.per = permission;
* }
* }
* let u1 = new User("meroo" , 20 , 200.45);
* // console.log(u1.getSalary());
* console.log(User.prototype);
* console.log(User.count);
* console.log(u1.msg());
* add(2000);
* Encapsulation
* polymorphism
  + Overriding
  + overloading
* Inheritance
  + A screenshot of a computer

    Description automatically generatedA screenshot of a computer

    Description automatically generated
  + A screenshot of a computer

    Description automatically generated
  + A screenshot of a computer

    Description automatically generated
  + A computer screen shot of a diagram

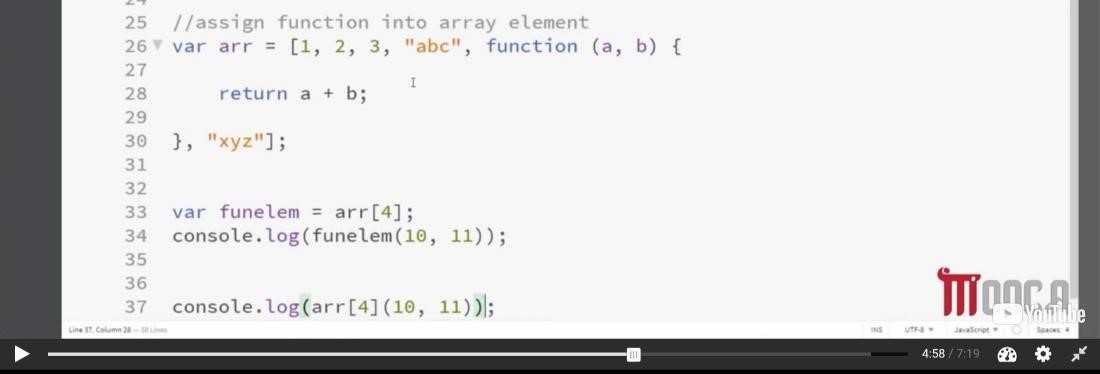
    Description automatically generated
  + Privileged Method -> getter

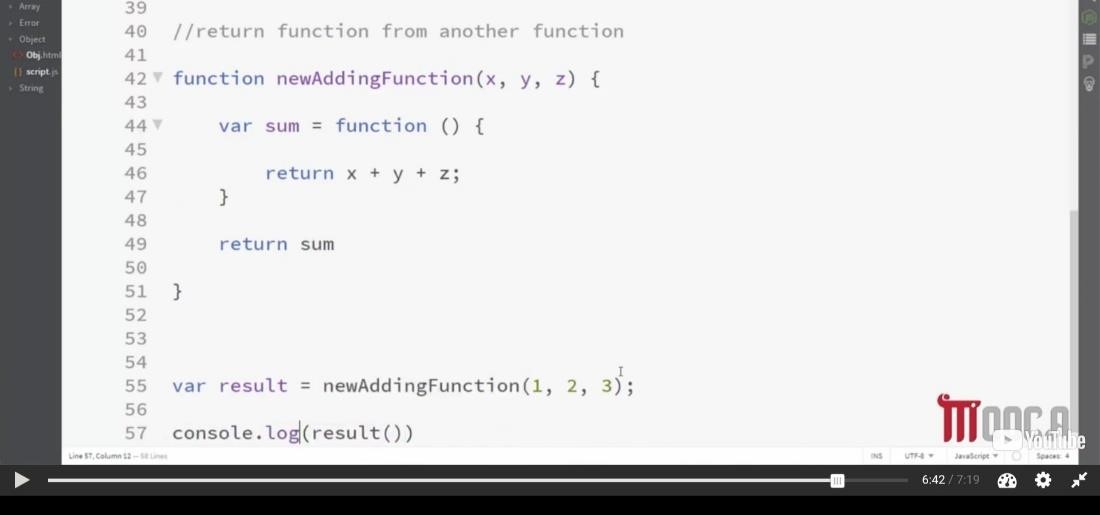
& setterA screenshot of a computer

Description automatically generated

* Constructor & factory Function (Function return object) || factory pattern
* class Std{
* student(name , age) { // constructor function
* this.name = name;
* this.age = age;
* };
* factory(n , a){ // factory function
* return {
* name :n,
* age : a
* }
* }
* }
* let students = new Std();
* console.log(students.factory("amira" , 20));
* That
* et Employee = function(n , d , a){
* this.name = n;
* this.depart = d;
* this.age = a;
* this.print = function(){
* var that = this;
* setTimeout(()=>{
* console.log(that.depart);
* } , 1000)
* }
* }
* let emp = new Employee("MoRaa" , "cs" , 20);
* emp.print();  //cs

## Custom Function

1. Statement & anonymous
   1. 
2. Assign fun into array element
   1. 
3. Return fun from another fun

a. 

## Special Operators

* Comma operators -> var first = “meroo” , last = “hassan”;
* Typeof -> Console.log(Typeof x);
* Ternary operator -> condition? Val1(if condition is true): val2(if condition is false)

## Communicating with the users -> Dialogue Box

* Getting data from the user -> Prompt (return string even number)
* Giving the user a pop up message -> Alert
* Ask the user a simple “yes or no” type of question -> confirm

## Notes

* F12
* <Script> inside body or head called internally
* Js parser -> translate code of js HTML parser -> translate code of HTML
* Conditional statement -> if ... else / switch
* Primitive data type -> o Undefined
  + null o Number o Boolean o String
* Operators -> o Binary operator
  + - Arithmetic -> x + y each of called operand
    - Assignment operators -> += , \*= , -= , /= , %=
    - Comparison operator -> < , > , =
    - Logical operators -> && , ||
* && , || use of short circle -> ex -> in case of && if first condition is false so final value is false
  + Ternary operator o Unary operator
    - Increment / decrement -> prefix / postfix
    - Logical operators -> ! (not) o False -> empty string, null, undefined, zero, false
  + && -> 3 && 5 && 7 (output is 7) because && is searching for value of false o Ex -> 3 && 0 && 7 && 8 (output is 0)
  + || -> 3 || 5 || 7 (output is 3) because || is searching for value of true o Ex -> 7 || 0 || 9 || 0(output is 7)
  + ! -> !9 (false) , !!9(true)
* Coercion -> convert the number to string / convert the Boolean to number o Var x = 5; var y = “5”; console.log(x == y); // true

(علشان لقى قيمه الواى باسترينج فحول قيمه الاكس لاسترنج) ▪

▪ Solution

* Strict equality -> ===
* 3<2<1
  + - 3<2 is false -> false < 1 -> 0 < 1 //true
    - Convert the Boolean to number
    - Solution
  + 3< (2<1) // false
  + Loop Statement o For() -> known the end
* For(in) -> associative array
* While() o Do while()
* Shadowing
  + Var z = 40 in global var z = 5 in local (in function) -> output of z = 40 ▪ even if print z after call function • Hoisted
  + What different between undeclared & uninitialized
    - Undeclared -> property value (other than function) are set to undefined until it is declared and set properly ex -> console.log(name) // name is not defined
    - Uninitialized -> variables are set to undefined by default xe -> let name ; name ? n : “undefined”
  + ***Link***
* <script src = “ ”></script>

## Note

***globalThis reference to window***

***Document.getElementById(“”).onclick = fun***

### ***(*لو الفانكشن بترجع حاجه يبقى احط الاقوا س غير كده لا*) >- ()*** •**علشان لو حطيت الاقواس والفانكشن مش بترجع فكده هترجع ديفاي ن*)*** •

***.(paramenter of function as event)preventDefault()***

***All Objects are Reference values -> obj1 = obj2 ; when you modify in obj1 the obj2 will be changed automatically***

***Array is object***

***All primitives are immutable***

let Employee = function(n , d , a){

        this.name = n;

        this.depart = d;

        this.age = a;

        this.print = function(){

                var that = this;

                setTimeout(()=>{

                        console.log(that.depart);

                } , 1000)

        }

}

let emp = new Employee("MoRaa" , "cs" , 20);

let emp1 = new Employee("MerOoo" , "is" , 19);

// emp = emp1;

// emp.depart = "it";

// console.log(emp);

// console.log(emp1);

for(i in emp){

        emp[i] = emp1[i];

}

emp.depart = "it";

console.log(emp);

console.log(emp1);

console.log(emp.\_\_proto\_\_); // prototype of constructor

IMPORTANT

let name = "meroo"; let age = 20;

function sum(str , ...values) {

        console.log(str);

        console.log(values);

}

sum `hello my name is ${name} and age is ${age}`;

After Course ->

1. javaScript as OOP
2. JS libraries as JQuery | DOJO
3. JS Unit Testing Frameworks MOKA | JASSMEN
4. JS frameworks angular | react vew.js

Remember

Export default class Person

Call Back

Transpiler -> to convert code with EC6 into vanile online

Can I Use -> website to know browser support version ( ES5 is famous in browser )

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Type Script \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Info about TypeScript
  + A screenshot of a computer

    Description automatically generated
  + 2012
  + TS = JS + Versions
  + Strongly typed -> select every variable depended on type of it (int || char & so on) while JS is Dynamically typed
  + Object oriented
  + Compiled language
  + Compilation Error
  + Is a superset of JS
  + Includes all version of JS
* CMD
  + –help (-h)
  + –watch (-w)
  + –target (-t) es5 || es6 -> default es3
  + –module (-m) -> external file
  + –removeComments
* DataType -> let variableName: type = value
  + Build-in Types
* // Number
* let age: number = 20;
* age = 25.5;
* // String
* let name1: string = "MoRaa"; // || 'MoRaa' || `MoRaa`
* let name2: string = 'MoRaa';
* let name3: string = `MoRaa`;
* let name4: string = `My Name is ${name3}`;
* // Boolean
* let isMarried = true;
* // function
* let fn: Function = () => console.log(`FUNCTION`);
* let fn1 = () => console.log(`FUNCTION`);
* // Object
* let obj:Object = {
* name : `Amira`,
* }
* let obj1 = {};
  + User defined Types
    - Class , enum , interface
* // class
* class Student { }
* // let std1: Student = null; // ERRORRRRRRRRR (:
* let std2: Student;
* // interface
* interface IFlyable {}
* let brid: IFlyable;
* // enum
* enum Color {}
* let co: Color;
  + Any Types
* let num: any = 7;
* // num = 'dwddew';
* // num = true;
* // num = {};
* // num = [];
  + OR
* let nb: number | boolean = 10;
* nb = false;
* // nb = {}; // ERRORRR
  + Return Types
* let greeting = (name: string): void => console.log(`${name}`);
* let greeting1 = (name: string): string => `${name}`;
* console.log(greeting(`MorAa`)); // undefined
* console.log(greeting1(`MerOoo`)); // MerOoo
  + Generic
* function g1<T>(arg:T):T{
* return arg;
* }
* let rg1 = g1<boolean>(false);
* let rga = g1<string>('hi');
  + Enum
* enum Color {red , green , blue};
* let color: Color = Color.blue;
* console.log(color); // 2
* enum Color1 {red = 'red' , green = 'green' , blue = 'blue'};
* let color1: Color1 = Color1.blue;
* console.log(color1); // blue
  + Interface
* interface IPerson{
* firstName : string,
* lastName : string,
* age : number,
* sayHello: ()=> string
* };
* let per1:IPerson = {
* firstName : "Amira",
* lastName : "Hassan",
* age : 20,
* sayHello: () => `Hola`
* };
* console.log(per1.firstName); // Amira
* console.log(per1.sayHello); // [Function: sayHello]
* console.log(per1.sayHello()); // Hola
  + - Import & Export
* // in test.ts //
* import {IPerson} from './personInterface';
* let per1:IPerson = {
* firstName : "Amira",
* lastName : "Hassan",
* age : 20,
* sayHello: () => `Hola`
* };
* console.log(per1.firstName); // Amira
* console.log(per1.sayHello); // [Function: sayHello]
* console.log(per1.sayHello()); // Hola
  + - // in personInterface.ts //
* export interface IPerson{
* firstName : string,
* lastName : strsing,
* age : number,
* sayHello: ()=> string
* };
  + Child can`t create object from parent
* class Parent{
* constructor(){
* console.log(`Constructor Of Parent`);
* }
* nameParent = String;
* }
* class Child1 extends Parent{
* constructor(){
* super();
* console.log(`Constructor Of Child1`);
* }
* nameChild1 = String;
* }
* class Child2 extends Child1{
* constructor(){
* super();
* console.log(`Constructor Of Child2`);
* }
* nameChild2 = String;
* }
* // let h1 = new Child1(); // Constructor Of Parent   Constructor Of Child1
* let ph1:Parent = new Child1(); // Constructor Of Parent   Constructor Of Child1
* let ph2:Parent = new Child2(); // Constructor Of Parent  Constructor Of Child1   Constructor Of Child2
* // let ch1:Child1 = new Parent();  // ERRORRRR
  + Overriding
* class Parent{
* msg(){
* console.log(`Parent`);
* }
* }
* class Child1 extends Parent{
* msg(){
* console.log(`Child1`);
* }
* }
* class Child2 extends Child1{
* }
* let ch1:Child1 = new Child1();
* ch1.msg();  // Child1
* let ch2:Child2 = new Child2();
* ch2.msg(); // Child1
* let pch2:Parent = new Child2();
* pch2.msg(); // Child1
* let pch1:Parent = new Child1();
* pch1.msg(); // Child1

GAME

* in cmd
  1. to be create npm GLOBAL
     + npm install –global typescript
  2. OR 🡺 to be create npm local
     + npm init (then) npm install –save-dev typescript
* create tsconfig.json
  1. press ctrl + space
* To create task.json
  1. ctrl + shift + p
  2. configure task
  3. tsc:build-tsconfig.json
* to watch result of change
  1. ctrl + shift + B -> build | w -> watch

NOTE

* assets -> includes images & audio & so on
* models ->